REMARKS

Claims 1-28 are pending in this application. Subsequent to this response, claims 1, 2, 12, 18, and 19 have been amended, and claims 13-17 have been canceled. No new matter is believed to be added by this response. Applicants submit that all of the claims are now in a form for allowance.

Claim Rejections – 35 U.S.C. § 102

In the Office action, claims 1, 2, 4-6 and 18, 19, 21-23 were rejected under 35 U.S.C. 102(b) as allegedly being anticipated by Judge et al. (US 6,430,570 – hereinafter, "Judge").

Claim 1 is an independent claim from which claims 2 and 4-6 depend either directly or indirectly. Claim 18 is an independent claim from which claims 19 and 21-23 depend either directly or indirectly. Independent claims 1 and 18 both require as elements determining which of a plurality of applications to effect removal from memory based on a received indication of application state.

Judge discloses an application manager for an imbedded device (col. 3, lines 9-10). Judge further discloses determining an execution state for applications:

> Application Manager 24 also allows the querying of application information such as which class objects 28a, 28b are currently loaded, which classes have application instances 30a, 30b, 30c, and what execution state each application instance is in (e.g., "initialized", "executing", "terminated"). (Col. 4, lines 55-59.)

However, Judge does not disclose determining whether to effect removal of any of the applications based on the received execution state. Judge makes decision on whether to remove applications from memory and to cache them based on the *class of the application*, not the execution state. The application class is defined as follows by Judge:

> Application Manager 24 is a server device operating as a network based class loader able to accept application class files 40 over network 14 via a network protocol 48 and bring the application classes contained in the application class files 40 to life. *As known by those skilled in the art, a class file 40 is a file comprising hardware architecture independent bytecodes generated by a Java.RTM. compiler.* The bytecodes are executable only by a JVM, such as JVM 22. Application Manager 24 is a non-typical network based ClassLoader, loading classes 40 from the network 14 on demand. Instead of loading class files 40

> through the Java.RTM. loadClass( ) method as normally done by typical class loaders, the Application Manager 24 instead waits on a server socket or other protocol 48 for a request to download class files 40. *When a download request is received, Application Manager 24 receives the class file 40 from the network 14, parses it into classes 28a, 28b, and then calls the Java.RTM. ClassLoader defineClass( ) method.* The class 28a, 28b should be resolved to ensure that it is ready for object instantiation by calling the Java.RTM. ClassLoader resolveClass( ) method. (Col. 3, line 57- col. 4, line 9, emphasis added in italics.)

The decision to remove the application from memory is based on class, not execution state. As disclosed by Judge, an application may become eligible for removal based on its application state, but the decision to remove an application from memory is determined by the application's class, and the need for memory. Therefore, at best Judge teaches removing applications from memory based on multiple factors (e.g., application state, application class, memory requirements, etc.), not making a determination based on a single indication of application state:

> Once executing, Application Manager 24 records the new running state of the application 26a, 26b, 26c. Each application class 28a, 28b includes a respective instance list 29a, 29b which identifies each instance that exists of its class. Each entry in the instance list 29a, 29b includes an execution state variable that is updated by Application Manager 24 during the lifecycle of the instant application 26a, 26b, 26c.
>
> Application Manager 24 handles all Java.RTM. exceptions or errors occurring in the application 26a, 26b, 26c which would otherwise cause the JVM 22 to terminate. Upon application termination, Application Manager 24 *records that the application 26a, 26b, 26c is no longer running (by updating the execution state corresponding to the instance identified in the instant list 29a, 29b), and causes the class object 28a, 28b to be garbage collected if the class 28a, 28b is not to be cached in memory 50.*
>
> *Caching of classes 28a, 28b is a unique feature of the invention that allows for better performance.* Instead of downloading the application class each and every time an application 26a, 26b, 26c is to be executed, the Application Manager 24 preferably caches the class object 28a, 28b by default and unloads the class 28a, 28b only when explicitly requested using the unloadAppl( ) method or when the memory management handler 27 selects the class to be unloaded as a result of a low- or no-memory condition. *In the preferred embodiment, the order in which currently loaded classes are unloaded by the memory management handler 27 is set using the setFreeAppsFirst( ) method. The setFreeAppsFirst( ) method allows a client to set or change the order in which applications are unloaded in case of a low- or no-memory condition. This may be accomplished in any of several ways, including by adding an unload priority field to each entry in the application list*

> *which indicates the ranking of the application for unloading the class.* By caching the class, the Application Manager 24 maintains a reference to the class, thereby forcing the Java Virtual Machine 22 not to garbage collect the class 28a, 28b. The application class 28a, 28b can then be re-used to create new instances 30a, 30b, 30c of the class 28a, 28b, and re-execute application functionality. (Col. 7, lines 12-51, emphasis added in italics.)

Further, in regard to the setFreeAppsFirst( ) method, Judge provides that decision s to unload memory are determined by class, not execution state:

> Application Manager 24 attempts to have applications 26a, 26b, 26c continue to execute in low- or no-memory situations. If, during the execution of an application 26a, 26b, 26c, the JVM 22 runs out of memory, an OutOfMemoryError error is generated. Application manager 24 includes a memory management handler 27 which handles low- or no-memory conditions. In the preferred embodiment, memory management handler 27 is triggered into action by the occurrence of an OutOfMemoryErorr generated by the JVM 22. In one embodiment, memory management handler 27 reacts by judiciously dumping class objects 28a, 28b and application objects 30a, 30b and 30c from its application cache. *The order for unloading cached class object 28a, 28b and application objects 30a, 30b and 30c is set using the setFreeAppsFirst( ) method. Alternatively, the order is determined according to a predetermined algorithm coded within the memory management handler 27 itself.* By calling the Java.RTM. Runtime.gc( ) method, Application Manager informs the JVM 22 that the unloaded class objects 28a, 28b and application objects 30a, 30b and 30c are available to be reclaimed. The unloading of class objects 28a, 28b from the application cache 52 forces the application objects 30a, 30b, 30c associated with the class 28a, 28b to be terminated and unloaded as well. Accordingly, if startup speed of application execution is important, it may be preferable to unload application objects 30a, 30b and 30c before unloading class objects 28a, 28b. *On the other hand, if it is important to keep a particular application running regardless, if may be preferable to unload all application objects* ***of another class*** *along with the class associated with those objects in order to free up enough memory to keep the particular application running.* Solutions to having applications terminate is to separately monitor memory levels reacting to low memory conditions, or to checkpoint application results allowing applications to be restarted continuing after a checkpoint. These approaches may allow executing applications to continue uninterrupted, or at least as close to uninterrupted as possible. (Col. 7, line 66 – col. 8, line 36, emphasis added in italics and bold.)

In no instance does Judge teach or disclose determining which of a plurality of applications to effect removal from memory based on a single received indication of application state. As such, Applicants submit that claims 1 and 18 are not anticipated by Judge. At best,

Judge teaches removing applications based on a plurality of indicators, application state and application class, not based on a single indicator of application state.

A proper rejection of a claim under 35 U.S.C. § 102 requires that a single prior art reference disclose each element of the claim. *See, e.g., W.L. Gore & Assoc., Inc. v. Garlock, Inc.*, 721 F.2d 1540, 220 USPQ 303, 313 (Fed. Cir. 1983). The test is the same for a process. Anticipation requires identity between the claimed process and a process of the prior art. The claimed process, including each step thereof, must have been described or embodied, either expressly or inherently, in a single reference. *See, e.g., Glaverbel S.A. v. Northlake Mkt'g & Supp., Inc.,* 45 F.3d 1550, 33 USPQ2d 1496 (Fed. Cir. 1995). Those elements must either be inherent or disclosed expressly. *See, e.g., Constant v. Advanced Micro-Devices, Inc.,* 848 F.2d 1560, 7 USPQ2d 1057 (Fed. Cir. 1988); *Verdegaal Bros., Inc. v. Union Oil Co.,* 814 F.2d 628, 2 USPQ2d 1051 (Fed. Cir. 1987). For anticipation, there must be no difference between the claimed invention and the reference disclosure, as viewed by a person of ordinary skill in the field of the invention. *See, e.g., Scripps Clinic & Res. Found. v. Genentech, Inc.,* 927 F.2d 1565, 18 USPQ2d 1001 (Fed. Cir. 1991). In summary, the single prior art reference must properly disclose, teach or suggest each element of the claimed invention. Moreover, "every element of the claimed invention must be literally present, arranged as in the claim. … The identical invention must be shown in as complete detail as is contained in the patent claim." *See, e.g., Richardson v. Suzuki Motor Company Co.* 868 F.2d 1226, 1236 (Fed. Cir. 1989).

As provided above, a claim can be anticipated only if each and every limitation of that claim is found in a single prior art reference. Judge teaches removing an application from memory based on multiple factors. Judge dos not teach, suggest or disclose determining which of the plurality of applications to effect removal from the memory *based on the received single indication* for each of the plurality of applications in memory, (emphasis added), as recited in amended claims 1 and 18. As such, Applicants respectfully submit that claims 1 and 18 are not anticipated by Judge and are in a form for allowance. Similarly, as claims 2, 4-6, 19, and 21-23 each depend directly or indirectly from their respective allowable base claims, Applicants submit that these claims are also in a form for allowance. See *In re Fine,* 5 U.S.P.Q.2d 1569, 1600 (Fed. Cir. 1988) ("Dependent claims are nonobvious under section 103 if the independent claims from which they depend are nonobvious."). Using this same rationale, dependent claims cannot be

anticipated if the independent claims from which they depend are not anticipated. Therefore, Applicants respectfully submit that claims 1, 2, 4-6, 18, 19 and 21-23 are in a form for allowance.

Claim Rejections – 35 U.S.C. § 103

Applicants first submit that, for a *prima facie* case of obviousness, the cited prior art references (when combined) "must teach or suggest all the claim limitations" MPEP § 2143. Thus, if the combination of references does not teach each of the claimed limitations, a finding of obviousness fails. In addition, the Patent Office has the burden under § 103 to establish a *prima facie* case of obviousness, which can be satisfied only by showing some objective teaching in the prior art would lead one to combine the relevant teachings of the references. *See In re Fine*, 837 F.2d 1071, 1074 (Fed. Cir. 1988). As such, an Applicant, to overcome an allegation of obviousness, can show that the cited prior art references (when combined) do not teach or suggest all the claim limitations or that there is not an objective teaching in the prior art that would lead one to combine the relevant teachings of the references.

A. Claims 3, 9, 10, 12, 20, 26, and 27

Claims 3, 9, 10, 12, 20, 26, and 27 were rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over Judge, as applied to claims 1 and 18 above, further in view of Enterprise JavaBeans Component Architecture: Designing and Coding Enterprise Applications, Henceforth referred to as "EJB." Claims 3, 9, and 10 depend either directly or indirectly from claim 1. Claim 12 is an independent method claim. Claims 20, 26 and 27 depend either directly or indirectly from claim 18.

As indicated above, Judge teaches removing an application from memory based on multiple factors. Judge dos not teach, suggest or disclose determining which of the plurality of applications to effect removal from the memory *based on the received single indication* for each of the plurality of applications in memory, (emphasis added), as recited in amended claims 1, 12, and 18. EJB does not correct this failing. Therefore, Judge in combination with EJB does not teach, suggest or make obvious determining which of a plurality of applications to effect removal from memory based on a received indication of application state as found in each of independent claims 1, 12 and 18 of the present application from which claims 3, 9, 10, 20, 26, and 27, directly

or indirectly depend. See *In re Fine,* 5 U.S.P.Q.2d 1569, 1600 (Fed. Cir. 1988) ("Dependent claims are nonobvious under section 103 if the independent claims from which they depend are nonobvious."). Each of claims 3, 9, 10, 20, 26, and 27 depend from an allowable base claim and are not made obvious by the combination of Judge in view of EJB. Applicants submit that claims 3, 9, 10, 12, 20, 26, and 27 are in a form for allowance.

Furthermore, Applicants maintain that the combination of Judge and EJB invokes hindsight reasoning. The Supreme Court has reaffirmed the *Graham* factors for determination of obvious under 35 U.S.C. 103(a). *KSR Int'l Co. v. Teleflex, Inc.* (*KSR*), No 04-1350 (U.S. Apr. 30, 2007). The four factual inquiries under *Graham* require examination of: (1) the scope and contents of the prior art; (2) the differences between the prior art and the claims in issue; (3) the level of ordinary skill in the pertinent art; and (4) the objective evidence of secondary consideration. *Graham v. John Deere* (*Graham*), 383 U.S. 1, 17-18, 149 USPQ 459, 467 (1966); see also 35 U.S.C. § 103 (2000).

The Court has further recognized that the requirement for a teaching, suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings, which was established by the Court of Customs and Patent Appeals, provides a helpful insight for determining whether the claimed subject matter is obvious under 35 U.S.C. § 103(a).

Where an invention is contended to be obvious based upon a combination of elements across different references, one should be able to identify particular reasons that would have prompted a person of ordinary skill in the relevant field to combine the [prior art] elements. *See, KSR Int'l Co.*, at 14, 15. This requirement prevents the use of "the inventor's disclosure as a blueprint for piecing together the prior art to defeat patentability—the essence of hindsight." *Ecolochem, Inc. v. So. Cal. Edison Co*., 227 F.3d 1361, 1371-72 (Fed. Cir. 2000) (quoting *In re Dembiczak*, 175 F.3d 994, 999 (Fed. Cir. 1999)).

Applicants submit that the current construction of the cited references in the manner provided in the Office Action requires hindsight reasoning, which the Federal Circuit has explicitly rejected. *See In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780, 1783 (Fed. Cir. 1992). As stated above, Judge teaches removing an application from memory based on multiple factors. Judge dos not teach, suggest or disclose determining which of the plurality of applications to

effect removal from the memory *based on the received single indication* for each of the plurality of applications in memory, (emphasis added), as recited in amended claims 1, 12 and 18, from which claims 3, 9, 10, 20, 26, and 27 depend. EJB does not correct these failings of Judge. It would not have been obvious to one of ordinary skill in the art to combine Judge in view of EJB, and the subject matter of the limitations not taught by either, to arrive at the presently claimed invention. Applicants earnestly request reconsideration, withdrawal of this rejection, and allowance of claims 3, 9, 10, 12, 20, 26, and 27.

B. Claims 7, 8, 11, 24, 25 and 28

Claims 7, 8, 11, 24, 25 and 28 were rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over Judge et al. (US 6,430,570 B1). Claims 7, 8 and 11 depend either directly or indirectly from independent claim 1; claims 24, 25 and 28 depend either directly or indirectly from independent claim 18.

The Office Action provides that official notice is taken that receiving an indication of a stateful state with no state record is common and well-known. Further, the Examiner references "Beginners Guide to Microsoft Word XP, Section 3" (undated reference) as teaching the limitation of receiving an indication of a stateful state with no state record.

Applicants respectfully traverse this rejection as there is no indication that "Beginners Guide to Microsoft Word XP, Section 3" is prior art to the claims of the present application and, as such, in accordance with MPEP 2144.03, Applicants hereby traverse these assertions of common knowledge or official notice unsupported by documentary evidence. However, assuming arguendo that "Beginners Guide to Microsoft Word XP, Section 3" is prior art to the claims of the present application, Judge teaches removing an application from memory based on multiple factors. Judge dos not teach, suggest or disclose determining which of the plurality of applications to effect removal from the memory *based on the received single indication* for each of the plurality of applications in memory, (emphasis added), as recited in amended claims 1 and 18. The official notice taken by the Examiner or "Beginners Guide to Microsoft Word XP, Section 3" (if it were prior art) does not correct this failing. Therefore, Judge in combination with The official notice taken by the Examiner or "Beginners Guide to Microsoft Word XP, Section 3" does not teach, suggest or make obvious determining which of a plurality of applications to effect removal from memory based on a received indication of application state as

found in each of independent claims 1 and 18 of the present application from which claims 7, 8, 11, 24, 25 and 28 directly or indirectly depend. See *In re Fine,* 5 U.S.P.Q.2d 1569, 1600 (Fed. Cir. 1988) ("Dependent claims are nonobvious under section 103 if the independent claims from which they depend are nonobvious."). Each of claims 7, 8, 11, 24, 25 and 28 depend from an allowable base claim and are not made obvious by the combination of Judge in view of official notice taken by the Examiner or "Beginners Guide to Microsoft Word XP, Section 3". Applicants therefore submit that claims 7, 8, 11, 24, 25 and 28 are in a form for allowance.

C. Claims 13-17

Claims 13-17 were rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over Judge et al. (US 6,430,570 B1), and further in view of Matsumoto (US 2002/0188461 A1). Applicants respectfully request to cancel claims 13-17, without prejudice.

Conclusion

In this response, claims 1, 2, 12, 18, and 19 have been amended and claims 13-17 canceled. Of the remaining pending claims, claims 1, 12, and 18 are independent claims. Since the Applicants respectfully assert that these independent claims are allowable, dependent claims 2-11 and 19-28 are also allowable. Thus, Applicants respectfully request allowance of all the pending claims.

No fee beyond the fee for the accompanying Request for Continued Examination (RCE) is believed due; however, the Commissioner is hereby authorized to charge any additional fees which may be required, or credit any overpayment to Deposit Account No. 14-0629.

Respectfully submitted,

__/David A. Cornett/_____
David A. Cornett
Registration No. 48,417

Customer No. 05642